Computer Vision Onboard UAVs for Civilian Tasks

Pascual Campoy · Juan F. Correa · Ivan Mondragón · Carol Martínez · Miguel Olivares · Luis Mejías · Jorge Artieda

Abstract Computer vision is much more than a technique to sense and recover environmental information from an UAV. It should play a main role regarding UAVs' functionality because of the big amount of information that can be extracted, its possible uses and applications, and its natural connection to human driven tasks, taking into account that vision is our main interface to world understanding. Our current research's focus lays on the development of techniques that allow UAVs to maneuver in spaces using visual information as their main input source. This task involves the creation of techniques that allow an UAV to maneuver towards features of interest whenever a GPS signal is not reliable or sufficient, e.g. when signal dropouts occur (which usually happens in urban areas, when flying through terrestrial urban canyons or when operating on remote planetary bodies), or when tracking or inspecting visual targets—including moving ones—without knowing their exact UMT coordinates. This paper also investigates visual servoing control techniques that use velocity and position of suitable image features to compute the references for flight control. This paper aims to give a global view of the main aspects related to the research field of computer vision for UAVs, clustered in four main active research lines: visual servoing and control, stereo-based visual navigation, image processing algorithms for detection and tracking, and visual SLAM. Finally, the results of applying these techniques in several applications are presented and discussed: this

Keywords UAV · Visual servoing · Image processing · Feature detection · Tracking · SLAM

1 Introduction

The vast infrastructure inspection industry frequently employs helicopter pilots and camera men who risk their lives in order to accomplish certain tasks, and taking into account that the way such tasks are done involves wasting large amounts of resources, the idea of developing an UAV—unmanned air vehicle—for such kind of tasks is certainly appealing and has become feasible nowadays. On the other hand, infrastructures such as oil pipelines, power lines or roads are usually imaged by helicopter pilots in order to monitor their performance or to detect faults, among other things. In contrast with those methods, UAVs appear as a cheap and suitable alternative in this field, given their flight capabilities and the possibility to integrate vision systems to enable them to perform otherwise human driven tasks or autonomous guiding and imaging.

Currently, some applications have been developed, among which we can find Valavanis' works on traffic monitoring path planning for multiple UAV cooperation and fire detection On the other hand, Ollero has also made some works with multi-UAVs. There are, too, some other works with mini-UAVs and vision-based obstacle avoidance made by Oh or by Serres . Moreover, Piegl and Valanavis in summarized the current status and future perspectives of the aforementioned vehicles. Applications where an UAV would manipulate its environment by picking and placing objects or by probing soil, among other things, can also be imagined and feasible in the future. In fact, there are plans to use rotorcraft for the exploration of planets like Mars

Additionally, aerial robotics might be a key research field in the future, providing small and medium sized UAVs as a cheap way of executing inspection functions, potentially revolutionizing the economics of this industry as a consequence. The goal of this research is to provide UAVs with the necessary technology to be visually guided by the extracted visual information. In this context, visual servoing techniques are applied in order to control the position of an UAV using the location of features in the image plane. Another alternative being explored is focused in the on-line reconstruction of the trajectory in the 3D space of moving targets (basically planes) to control the UAV's position

Vision-based control has become interesting because machine vision directly detects a tracking error related to the target rather than indicating it via a coordinate system fixed to the earth. In order to achieve the aforemention detection, GPS is used to guide the UAV to the vicinity of the structure and line it up. Then, selected or extracted features in the image plane are tracked. Once features are detected and tracked, the system uses the image location of these features to generate image-based velocity references to the flight control.

In the following section briefly describe the different components that are needed to have an UAV ready to flight, and to test it for different applications. Section 3,

explains with details the different approaches to extract useful information to achieve visual servoing in the image plane based on features and on appearance. Some improvements in 3D motion reconstruction are also pointed out. Section 4 describes visual control schemes employed to aim visual servoing, and the particular configuration of the control system assigned to close the visual control loop. Section 5 deals with the stereo configuration and theory to make motion and height estimation based on two views of a scene. Next, in Section 6 the simultaneous localization and Mapping problem based on visual information is addressed, with particular emphasis on images taken from an UAV. Section 7 shows experimental results of different applications, and Section 8, finally, deals with conclusions and future work.

2 System Overview

Several components are necessary to complete an operational platform equipped with a visual system to control UAVs. It is a multidisciplinary effort that encloses different disciplines like system modeling and control, data communication, trajectory planning, image processing, hardware architecture, software engineering, and some others. All this knowledge is traduced into an interconnected architecture of functional blocks. The Computer Vision Group at UPM has three fully operational platforms at its disposal, whereas two of them are gas powered Industrial Twim 52 c.c helicopters producing about 8 hp, which are equipped with an AFCS helicopter flight controller, a guidance system, a Mini Itx onboard vision computer, and an onboard 150 W generator. These helicopters are used for outdoors applications, as shown in Fig. 1, where one of the powered gas platforms performs an experimental autonomous flight. The third platform is a Rotomotion SR20 UAV with an electric motor of 1,300 W, 8A. It also has a Nano Itx onboard vision computer and WiFi ethernet for telemetry data. It is used on indoors and outdoors applications. In this section, a description of the main modules, their structure and some basic

Fig. 1 Aerial platform COLIBRI while is performing an experimental detection and tracking of external visual references

functionalities is provided. In general terms, the whole system can be divided into two components:

1. An onboard subsystem composed by:
   - Vision computer with the image processing algorithms and related image treatment programs.
   - Flight computer with Flight control software.
   - Cameras.
   - Communication interface with flight control and with ground subsystem.
2. A ground subsystem:
   - Ground computer for interaction with the onboard subsystem, and data analysis.
   - Communication interface.
   - Data storage.

Those components' division can be reorganized into subsystems, which are described below.

2.1 Flight Control Subsystem

Most complex physical systems' dynamics are nonlinear. Therefore, it is important to understand under which circumstances a linear modeling and control design will be adequate to address control challenges. In order to obtain a linear dynamic model, the hover state can be used as a point of work to approximate the helicopter dynamics by linear equations of motion. Using this approximation, linearization around this state gives a wide enough range of linearity to be useful for controlling purposes.

The control system is based on single-input single-output (SISO) proportional-integral-derivative (PID) feedback loops. Such a system has been tested to provide basic sufficient performance to accomplish position and velocity tracking near hover flight The advantage of this simple feedback architecture is that it can be implemented without a model of the vehicle dynamics (just kinematic), and all feedback gains can be turned on empirically in flight. The performance of this type of control reaches its limits when it is necessary to execute fast and extreme maneuvers. For a complete description of the control architecture, refer to

The control system needs to be communicated with external processes (Fig. 2) in order to obtain references to close external loops (e.g. vision module, Kalman filter for state estimation, and trajectory planning). The communication is made through a high level layer that routes the messages to the specific process. The next subsection introduces the communication interface in detail.

2.2 Communication Interface

A client-server architecture has been implemented based on TCP/UDP messages, allowing embedded applications running on the computer onboard the autonomous helicopter to exchange data between them and with the processes running on the ground station. The exchange is made through a high level layer which routes the messages to the specific process. Switching and routing a message depends on the type of information received. For example, the layer can switch between position

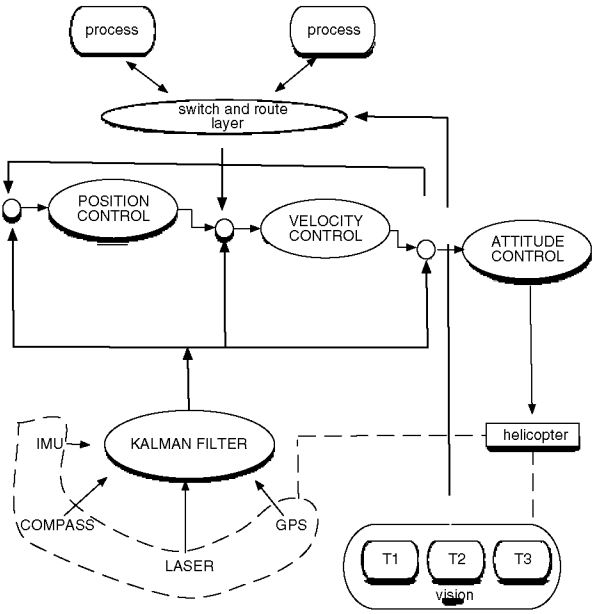


Fig. 2 Control system interacting with external processes. Communication is made through a high level layer using specific messages routed for each process

and velocity control depending on the messages received from an external process. The mechanism used for this purpose consists in defining data structures containing a field that uniquely identifies the type of information and the destination of a message. Some of the messages defined for flight control are: velocity control, position control, heading, attitude and helicopter state.

Figure 3, shows a case in which two processes are communicating through the switching layer. One process is sending commands to the flight control (red line), while the other one (blue line) is communicating with another process.

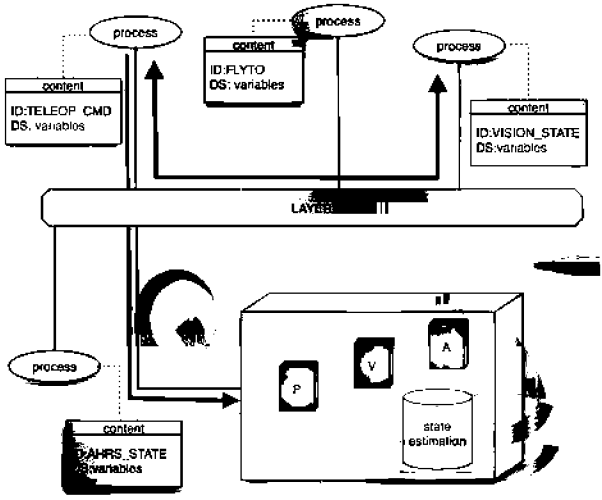


Fig. 3 Switching Layer. TCP/UDP messages are used to exchange data between flight controller and other process. Exchange is driven by a high level layer which routes the data to the specific process

2.3 Visual Subsystem

The visual subsystem is a compound of a servo controlled Pan Tilt platform, an on-board computer and a variety of cameras and visual sensors, including analog/digital cameras (Firewire, USB, IP-LAN and other digital connections), with the capability of using configurations based on single, stereo cameras, arrays of synchronous multiple sensor heads and many other options. Additionally, the system allows the use of Gimbals' platforms and other kinds of sensors like IF/UV spectrum cameras or Range Finders. Communication is based on a long-range wireless interface which is used to send images for ground visualization of the onboard view and for visual algorithm supervision. Applications and approaches designed to perform visual tasks encompass optical flow, Hough transform, camera calibration, stereo vision to corner detection, visual servoing control implementation and Kalman filtering, among others.

Scene information obtained from image processing and analysis provides data related to the camera's coordinate system. This information is useful for purposes of automatic camera control, but not for the attitude and position control of the UAV. This issue is solved by fixating and aligning the camera's frame reference with the vehicle body-frame. Next section enumerates some basic algorithms of visual information extracted for controlling purposes.

3 Visual Tracking

The main interest of the computer vision group at UPM is to incorporate vision systems in UAVs in order to increase their navigation capabilities. Most of this effort is based on image processing algorithms and tracking techniques that have been implemented in UAVs and will be described below.

3.1 Image Processing

Image processing is used to find characteristics in the image that can be used to recognize an object or points of interest. This relevant information extracted from the image (called features) ranges from simple structures, such as points or edges, to more complex structures, such as objects. Such features will be used as reference for the visual flight control.

Most of the features used as reference are interest points, which are points in an image that have a well-defined position, can be robustly detected, and are usually found in any kind of images. Some of these points are corners formed by the intersection of two edges, and some others are points in the image whose context has rich information based on the intensity of the pixels. A detector used for this purpose is the Harris Corner detector [18]. It extracts a lot of corners very quickly based on the magnitude of the eigenvalues of the autocorrelation matrix. However, it is not enough to use this measure in order to guarantee the robustness of the corner, since the purpose of the features' extraction is to track them along an image sequence. This means that good features to track have to be selected in order to ensure the stability of the tracking process. The robustness of a corner extracted with the Harris

detector can be measured by changing the size of the detection window, which is increased to test the stability of the position of the extracted corners. A measure of this variation is then calculated based on a maximum difference criteria. Besides, the magnitude of the eigenvalues is used to only keep features with eigenvalues higher than a minimum value. Combination of such criteria leads to the selection of the good features to track.

Another widely used algorithm is the SIFT (scale invariant feature transform) detector of interest points, which are called keypoints in the SIFT framework. This detector was developed with the intention to use it for object recognition. Because of this, it extracts keypoints invariant to scale and rotation using the gaussian difference of the images in different scales to ensure invariance to scale. To achieve invariance to rotation, one or more orientations based on local image gradient directions are assigned to each keypoint. The result of all this process is a descriptor associated to the keypoint, which provides an efficient tool to represent an interest point, allowing an easy matching against a database of keypoints. The calculation of these features has a considerable computational cost, which can be assumed because of the robustness of the keypoint and the accuracy obtained when matching these features. However, the use of these features depends on the nature of the task: whether it needs to be done fast or accurate.

The use of other kind of features, such as edges, is another technique that can be applied on semi-structured environments. Since human constructions and objects are based on basic geometrical figures, the Hough transform becomes a powerful technique to find them in the image. The simplest case of the algorithm is to find straight lines in an image that can be described with the equation $y = mx + b$. The main idea of the Hough transform is to consider the characteristics of the straight line not as image points x or y, but in terms of its parameters m and b. The procedure has more steps to re-parameterize into a space based on an angle and a distance, but what is important is that if a set of points form a straight line, they will produce sinusoids which cross at the parameters of that line. Thus, the problem of detecting collinear points can be converted to the problem of finding concurrent curves. To apply this concept just to points that might be on a line, some pre-processing algorithms are used to find edge features, such as the Canny edge detector or the ones based on derivatives of the images obtained by a convolution of image intensities and a mask These methods have been used in order to find power lines and isolators in an inspection application

3.2 Feature Tracking

The problem of tracking features can be solved with different approaches. The most popular algorithm to track features like corner features or interest points in consecutive images is the Lukas–Kanade algorithm It works under two premises: first, the intensity constancy in the vicinity of each pixel considered as a feature; secondly, the change in the position of the features between two consecutive frames must be minimum, so that the features are close enough to each other. Given these conditions to ensure the performance of the algorithm, it can be expressed in the following form: if we have a feature position $p_i = (x, y)$ in the image I_k, the objective of the tracker is to find the position of the same feature in the image I_{k+1} that fits the expression $p_i' = (x, y) + t$, where $t = (t_x, t_y)$. The t vector is known as

the optical flow, and it is defined as the visual velocity that minimizes the residual function $e(t)$ defined as:

$$e(t) = \sum^{W} (I_k(p_i) - I_{k+1}(p_i + t))^2 w(W) \tag{1}$$

where $w(x)$ is a function to assign different weights to comparison window W. This equation can be solved for each tracked feature, but since it is expected that all features on physical objects move solidary, summation can be done over all features. The problem can be reformulated to make it possible to be solved in relation to all features in the form of a least squares' problem, having a closed form solution. In Section 3.3 more details are given. Whenever features are tracked from one frame to another in the image, the measure of the position is affected by noise. Hence, a Kalman filter can be used to reduce noise and to have a more smooth change in the position estimation of the features. This method is also desirable because it provides an estimation of the velocity of the pixel that is used as a reference to the velocity flight control of the UAV.

Another way to track features is based on the rich information given by the SIFT descriptor. The object is matched along the image sequence comparing the model template (the image from which the database of features is created) and the SIFT descriptor of the current image, using the nearest neighbor method. Given the high dimensionality of the keypoint descriptor (128), its matching performance is improved using the Kd-tree search algorithm with the Best Bin First search modification proposed by Lowe The advantage of this method lies in the robustness of the matching using the descriptor, and in the fact that this match does not depend on the relative position of the template and the current image. Once the matching is performed, a perspective transformation is calculated using the matched Keypoints, comparing the original template with the current image. Then, the RANSAC algorithm is applied to obtain the best possible transformation, taking into consideration bad correspondences. This transformation includes the parameters for translation, rotation and scaling of the interest object, and is defined in Eqs. 2 and 3.

$$X_k = HX_0 \tag{2}$$

$$\begin{pmatrix} x_k \\ y_k \\ \lambda \end{pmatrix} = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & 1 \end{pmatrix} \begin{pmatrix} x_0 \\ y_0 \\ 1 \end{pmatrix} \tag{3}$$

where $(x_k, y_k, \lambda)^T$ is the homogeneous position of the matched keypoint against $(x_0, y_0, 1)^t$ position of the feature in the template image, and H is the homography transformation that relates the two features. Considering that every pair of matched keypoints gives us two equations, we need a minimum of four pairs of correctly matched keypoints to solve the system. Keeping in mind that not every match may be correct, the way to reject the outliers is to use the RANSAC algorithm to robustly estimate the transformation H. RANSAC achieves its goal by iteratively selecting a random subset of the original data points, testing it to obtain the model and then evaluating the model consensus, which is the total number of original data points that best fit the model. This procedure is then repeated a fixed number of times, each time producing either a model which is rejected because too few points are

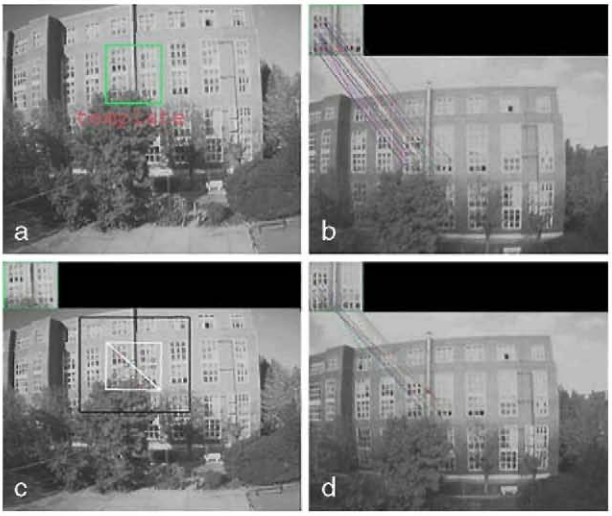


Fig. 4 Experiments with planar objects in order to recover the full pose of the tracked object using SIFT. In the sub-figure **a** a template is chosen from the initial frame. In **b** the SIFT database is generated using the extracted keypoints. In **c** points are searched in a region twice the size of the template in the next image using the previous position as initial guess. **d** Subfigure shows the matching achieved by the tracking algorithm

classified as inliers or a model that better represents the transformation. If the total trials are reached, a good solution can not be obtained. This situation enforces the correspondences between points from one frame to another. Once a transformation is obtained, the pose of the tracked plane can be recovered using the information in the homography. Figure 4 shows an implementation of this method.

3.3 Appearance Based Tracking

Tracking based on appearance does not use features. On the other hand, it uses a patch of pixels that corresponds to the object that wants to be tracked. The method to track this patch of pixels is the same L–K algorithm. This patch is related to the next frame by a warping function that can be the optical flow or another model of motion. The problem can be formulated in this way: lets define X as the set of points that forms the template image $T(\mathbf{x})$, where $\mathbf{x} = (x, y)^T$ is a column vector with the coordinates in the image plane of the given pixel. The goal of the algorithm is to align the template $T(\mathbf{x})$ with the input image $I(\mathbf{x})$. Because $T(\mathbf{x})$ is a sub-image of $I(\mathbf{x})$, the algorithm will find the set of parameters $\mu = (\mu_1, \mu_2, ...\mu_n)$ for motion model function $W(\mathbf{x}; \mu)$, also called the warping function. The objective function of the algorithm to be minimized in order to align the template and the actual image is Eq. 4

$$\sum_{\forall \mathbf{x} \in X} (I(W(\mathbf{x}; \mu) - T(\mathbf{x}))^2 \tag{4}$$

Since the minimization process has to be made with respect to μ, and there is no lineal relation between the pixel position and its intensity value, the Lukas–Kanade algorithm assumes a known initial value for the parameters μ and finds increments of the parameters $\delta\mu$. Hence, the expression to be minimized is:

$$\sum_{\forall \mathbf{x} \in X} (I(W(\mathbf{x}; \mu + \delta\mu) - T(\mathbf{x}))^2 \tag{5}$$

and the parameter actualization in every iteration is $\mu = \mu + \delta\mu$. In order to solve Eq. 5 efficiently, the objective function is linearized using a Taylor Series expansion employing only the first order terms. The parameter to be minimized is $\delta\mu$. Afterwards, the function to be minimized looks like Eq. 6 and can be solved like a "least squares problem" with Eq. 7.

$$\sum_{\forall \mathbf{x} \epsilon X} \left(I(W(\mathbf{x}; \mu) + \nabla I \frac{\partial W}{\partial \mu} \delta\mu - T(\mathbf{x}) \right)^2 \tag{6}$$

$$\delta\mu = H^{-1} \sum_{\forall \mathbf{x} \epsilon X} \left(\nabla I \frac{\partial W}{\partial \mu} \right)^T (T(\mathbf{x}) - I(W(\mathbf{x}; \mu)) \tag{7}$$

where H is the Hessian Matrix approximation,

$$H = \sum_{\forall \mathbf{x} \epsilon X} \left(\nabla I \frac{\partial W}{\partial \mu} \right)^T \left(\nabla I \frac{\partial W}{\partial \mu} \right) \tag{8}$$

More details about this formulation can be found in and where some modifications are introduced in order to make the minimization process more efficient, by inverting the roles of the template and changing the parameter update rule from an additive form to a compositional function. This is the so called ICA (Inverse Compositional Algorithm), first proposed in These modifications where introduced to avoid the cost of computing the gradient of the images, the Jacobian of the Warping function in every step and the inversion of the Hessian Matrix that assumes the most computational cost of the algorithm.

Besides the performance improvements that can be done to the algorithm, it is important to explore the possible motion models that can be applied to warp the patch of tracked pixels into the $T(\mathbf{x})$ space, because this defines the degrees of freedom of the tracking and constrains the possibility to correctly follow the region of interest. Table 1 summarizes some of the warping functions used and the degrees of freedom. Less degrees of freedom make the minimization process more stable and accurate, but less information can be extracted from the motion of the object. If a perspective transformation is applied as the warping function, and if the selected patch corresponds to a plane in the world, then 3D pose of the plane can be

Table 1 Warping functions summary

Name	Rule	D.O.F
Optical flow	$(x, y) + (t_x, t_y)$	2
Scale+translation	$(1+s)((x, y) + (t_x, t_y))$	3
Scale+rotation+ translation	$(1+s)(R_{2x2}(x, y)^T + (t_x, t_y)^T)$	4
Affine	$\begin{pmatrix} 1+\mu_1 & \mu_3 & \mu_5 \\ \mu_2 & 1+\mu_4 & \mu_6 \end{pmatrix}$	6
Perspective	$\begin{pmatrix} \mu_1 & \mu_2 & \mu_3 \\ \mu_2 & \mu_5 & \mu_6 \\ \mu_7 & \mu_8 & 1 \end{pmatrix}$	8

(a) (b)

Fig. 5 Experiments using appearance based tracking were conducted to track a template in the scene. **a** Is the initial frame of the image sequence. Image region is manually selected and tracked along image sequence, using a scale + translation model (see Table 1). **b** Shows the tracked template 50 frames later from image (**a**). *Sub-images in the bottom of each figure* represent the initial template selected and the warped patch transformed into the template coordinate system

reconstructed from the obtained parameters. Figure 5 shows some tests carried out using a translation+scale motion model.

4 Visual Flight Control

4.1 Control Scheme

The flight control system is composed of three control loops arranged in a cascade formation, allowing it to perform tasks in different levels depending on the workspace of the task. The first control loop is in charge of the attitude of the helicopter. It interacts directly over the servomotors that define the four basic variables: cyclic/collective pitch of the principal rotor, cyclic/collective pitch of the tale rotor, longitudinal cyclic pitch, and the latitudinal cyclic pitch. The kinematic and dynamic models of the helicopter relate those variables with the six degrees of motion that this kind of vehicle can have in the cartesian space. As mentioned above in Section 2.1, the hover state can be used as a point of work to approximate the helicopter's dynamics by linear equations of motion. Using this approximation, linearization around this state gives a wide enough range of linearity that is useful for control purposes. For this reason, this control is formed of decoupled PID controllers for each of the control variables described above.

The second controller is a velocity-based control responsible of generating the references for the attitude control. It is implemented using a PI configuration. The controller reads the state of the vehicle from the state estimator and gives references to the next level, but only to make lateral and longitudinal displacements. The third controller (position based control) is at the higher level of the system, and is designed to receive GPS coordinates.The control scheme allows different modes of operation, one of which is to take the helicopter to a desired position (position control). Once the UAV is hovering, the velocity based control is capable of receiving references to keep the UAV aligned with a selected target, and it leaves the stability of the aircraft to the most internal loop in charge of the attitude. Figure 6 shows the structure of the

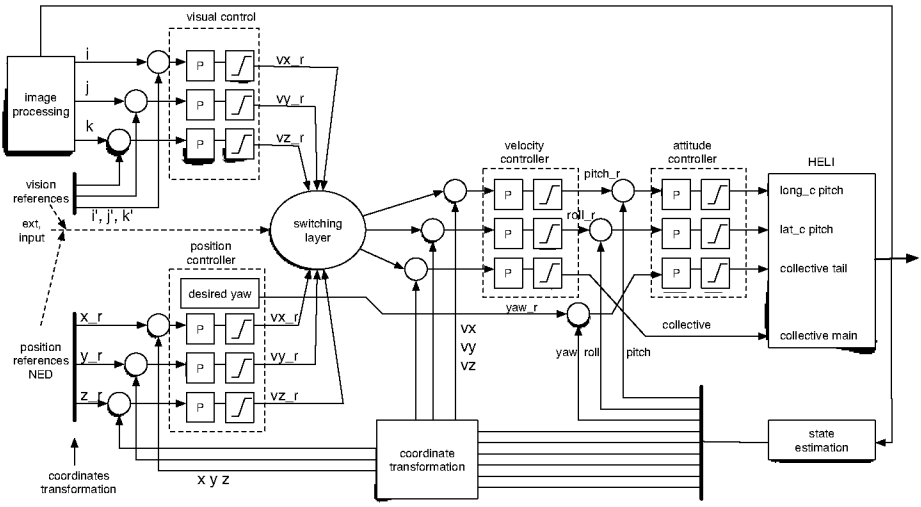


Fig. 6 Schematic flight control system. The inner velocity control loop is made of three cascade decoupled PID controllers. The outer position control loop can be externally switched between the visual based controller and the GPS based position controller. The former can be based on direct feature visual control or alternatively on visual estimated world positioning

flight control system with more details, and the communication interface described in Section 2.2, that is the key to integrate the visual reference as an external loop. Next subsection describes how this has been achieved.

4.2 Visual References Integration

The first step to design the control task in the image coordinates is to define the camera's model and the dynamics of a feature in the image, in order to construct a control law that properly represents the behavior of the task. Figure 7 shows the basic PinHole model of the camera, where $P^c(x, y, z)$ is a point in the camera coordinates system, and $p^c(i, j)^T$ denotes the projection of that point in the image plane π. Velocity of the camera can be represented with the vector $V = (v_x^c, v_y^c, v_z^c)^T$, while vector $\omega = (w_x^c, w_y^c, w_z^c)^T$ depicts the angular velocity. Considering that objects in the scene don't move, the relative velocity of a point in the world related to the camera's optical center can be expressed in this form:

$$\dot{P}^c = -\left(V + \omega \times P^c\right) \tag{9}$$

Using the well known Eq. 10 based on the camera calibration matrix that expresses the relationship between a point in the camera's coordinate system and its projection in the image plane, deriving Eq. 10 with respect to time, and replacing Eq. 9, it is possible to obtain a new Eq. 11 that describes a differential relation between the

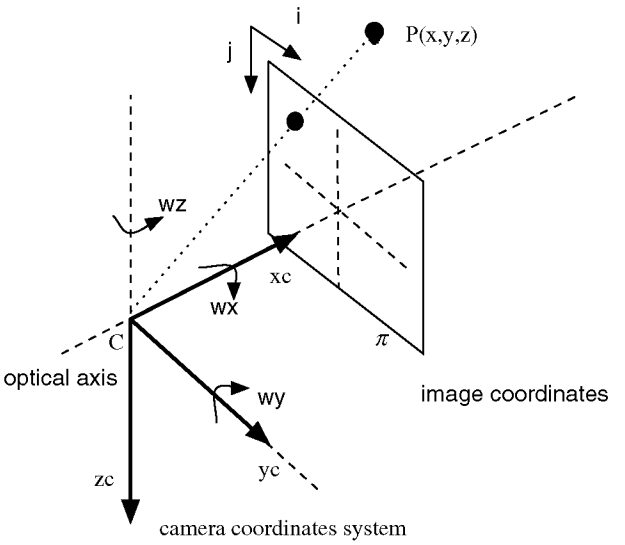


Fig. 7 PinHole camera model to describe the dynamic model, where $P(x, y, z)$ is a point in the camera coordinates system, $p(i, j)^T$ represents the projection of that point in the image plane π and the vector $\omega = (w_x, w_y, w_z)^T$ is the angular velocity

velocity of the projection of a point in the image and the velocity vector of the camera V and ω.

$$p^c = \mathbf{K}P^c \tag{10}$$

$$\dot{p^c} = -\mathbf{K}\left(V + \omega \times P^c\right) \tag{11}$$

Since the visual servoing task is designed to only make lateral and longitudinal displacements, and the camera is fixed looking to the front, it is possible to assume that the angular velocity is despicable because of the short range of motion of the pitch angles and the velocity constraint imposed to the system. Hence, Eq. 11 is reduced to this expression:

$$\dot{p^c} = \begin{bmatrix} \frac{di}{dt} \\ \frac{dj}{dt} \end{bmatrix} = - \begin{bmatrix} \frac{f}{x^c} & 0 \\ 0 & \frac{f}{x^c} \end{bmatrix} \begin{bmatrix} v_x^c \\ v_z^c \end{bmatrix} \tag{12}$$

This expression permits the introduction of the references described in Section 3 as a single measure, using the center of mass of the features or the patch tracked by the image processing algorithm, and using the velocity control module of the Flight Control System described above in this section.

5 Stereo Vision

This section shows a system to estimate the altitude and motion of an aerial vehicle using a stereo visual system. The system first detects and tracks interest points in the scene. The depth of the plane that contains the features is calculated matching

features between left and right images and then using the disparity principle. Motion is recovered tracking pixels from one frame to the next one, finding its visual displacement and resolving camera rotation and translation by a least-square method

5.1 Height Estimation

Height Estimation is performed on a stereo system using a first step to detect features in the environment with any of the technique mentioned in Section 3. This procedure is performed in each and every one of the stereo images.

As a second step, a correlation procedure is applied in order to find the correspondences between the two sets of features from the right and left images. Double check is performed by checking right against left, and then comparing left with right. The correlation stage is based on the ZNNC—zero mean normalized cross correlation—which offers good robustness against light and environmental changes

Once the correspondence has been solved, considering an error tolerance, given that the correspondence is not perfect, and thanks to the fact that all pixels belong to the same plane, the stereo disparity principle is used to find the distance to the plane that contains the features. Disparity is inversely proportional to scene depth multiplied by the focal length (f) and baseline (b). The depth is computed using the expression for Z shown in Fig. 8.

Figure 9 shows the algorithm used to estimate the distance from the stereo system to the plane. In the helicopter, the stereo system is used in two positions. In the first one, the stereo system is looking down, perpendicular to ground, so that the estimated distance corresponds to the UAV altitude. In the second configuration, the stereo system is looking forward, and by so doing the estimated distance corresponds to the distance between the UAV and an object or feature.

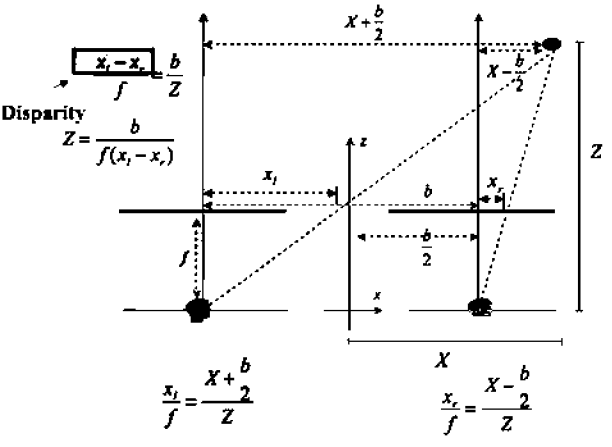


Fig. 8 Stereo Disparity for aligned cameras with all pixel in the same plane. Stereo disparity principle is used to find the distance to the plane that contains the features

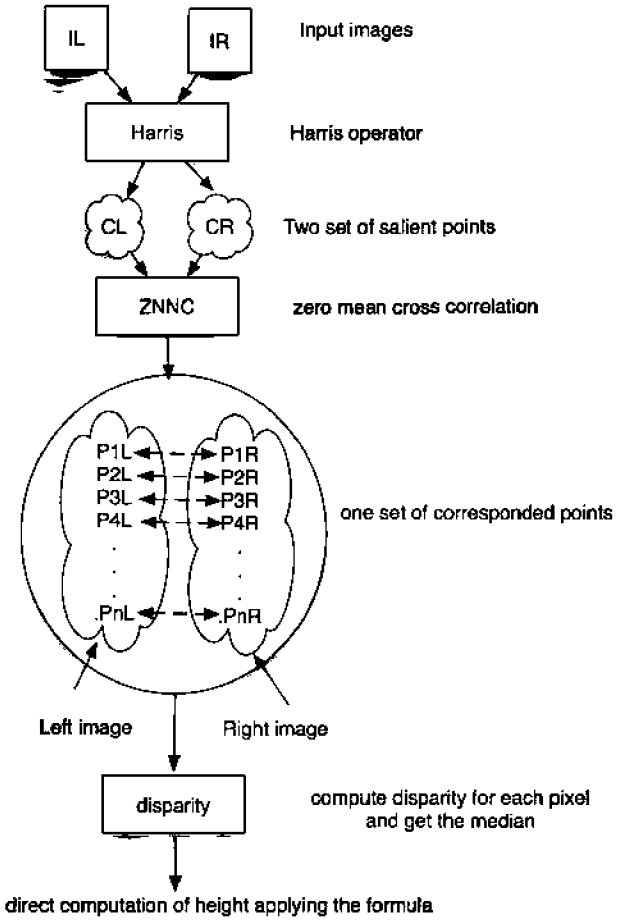


Fig. 9 Height estimation using the Harris Corner detector and ZNNC. Height is obtained employing the stereo disparity principle

5.2 Motion Estimation

Motion estimation is performed using at a first stage the same technique used for feature correspondence between left and right corners: the zero mean normalized cross-correlation (ZNNC). Correlation is performed within a certain pixel distance from each other keeping those points in a correlation coefficient higher than 0.85. The motion problem estimation is done aligning two sets of points whose correspondence is known, and finding the rotation matrix and translation vector, i.e, 3D transformation matrix T that minimizes the mean-squares' objective function $\min_{R,t} \sum_N \parallel TP_{k-1} - P_k \parallel^2$. Problem can be solved using Iterative Closes Point (ICP) registration and motion parameter estimation using SVD. Assuming there are two sets of points which are called data and model: $P = \{p_i\}_1^{N_p}$ and $M = \{m_i\}_1^{N_m}$ respectively with $N_p \neq N_m$, whose correspondence is known. The problem is how to compute the rotation (R) and translation (t) producing the best possible alignment of P and M by relation them with the equation $M = RP + t$. Lets define the closest

point in the model to a data point p as $cp(p) = \arg\ \min_{m \in M} \parallel m - p \parallel$. Then, the ICP step goes like this:

1. Compute the subset of closest points (CP) , $y = \{m \in M \mid p \in P : m = cp(p)\}$
2. Compute the least-squares estimate of motion bringing P onto y: $(R, t) = \arg\ \min_{R,t} \sum_{i=1}^{N_p} \parallel y_i - Rp_i - t \parallel^2$
3. Apply motion to the data points, $P \leftarrow RP + t$
4. If the stopping criterion is satisfied, exit; else goto 1.

Calculating the rotation and the translation matrix using SVD can be summarized as follows: first, the rotation matrix is calculated using the centroid of the set of points. Centroid is calculated as $y_{c_i} = y_i - \bar{y}$ and $p_{c_i} = p_i - \bar{p}$, where $\bar{y} = \frac{1}{N_p} \sum_{N_p} cp(p_i)$ and $\bar{p} = \frac{1}{N_p} \sum_{N_p} p_i$. Then, rotation is found minimizing $\min_R \sum_{N_p} \parallel y_{c_i} - Rp_{c_i} \parallel^2$. This equation is minimized when trace (RK) is maximized with $K = \sum_{N_p} y_{c_i} p_{c_i}^T$. Matrix K is calculated using SVD as $K = VDU^T$. Thus, the optimal rotation matrix that maximizes the trace is $R = VU^T$. The optimal translation that aligns the centroid is $t = \bar{y} - P\bar{p}$.

Section 7.3, shows tests and applications' development using the stereo system and algorithms explained in this section.

6 Airborne Visual SLAM

This section presents the implementation of an aerial visual SLAM algorithm with monocular information. No prior information of the scene is needed for the proposed formulation. In this approach, no extra absolute or relative information, GPS or odometry are used. The SLAM algorithm is based on the features or corners' matching process using SURF features or on the Harris Corner detector
First, the formulation of the problem will be described. Then, the details of the Kalman filter will be explained and, finally, this section will end with the description of this approach's particularities.

6.1 Formulation of the Problem

The problem is formulated using state variables to describe and model the system. The state of the system is described by the vector:

$$X = [\mathbf{x}, \mathbf{s}_1, \mathbf{s}_2, \mathbf{s}_3, ...] \tag{13}$$

where $\mathbf{x}$ denotes the state of the camera and $\mathbf{s}_i$ represents the state of each feature. Camera state has 12 variables. The First six variables represent the position of the vehicle in iteration k and in the previous iteration. The Next six variables, vector $[p, q, r]$, represent the rotation at iteration k and k-1. Rotation is expressed using

Rodrigues' notation, which expresses a rotation around a vector with the direction of $\omega = [p, q, r]$ of an angle $\theta = \sqrt{p^2 + q^2 + r^2}$. The rotation matrix is calculated from this representation using

$$e^{\tilde{\omega}\theta} = I + \tilde{\omega}\sin(\theta) + \tilde{\omega}^2(1 - \cos(\theta)) \tag{14}$$

where I is the 3×3 identity matrix and $\tilde{\omega}$ denotes the antisymmetric matrix with entries

$$\tilde{\omega} = \begin{bmatrix} 0 & -r & q \\ r & 0 & -p \\ -q & p & 0 \end{bmatrix} \tag{15}$$

Therefore the state of the camera, not including the features, is composed by the following 12 variables,

$$\mathbf{x} = [x_k, x_{k-1}, y_k, y_{k-1}, z_k, z_{k-1}, p_k, p_{k-1}, q_k, q_{k-1}, r_k, r_{k-1}] \tag{16}$$

Another implementation of monocular SLAM uses quaternion to express the rotation [31]. The use of Rodrigues' notation, instead of quaternion, allows the reduction of the problem's dimension by only using three variables to represent the rotation.

Each feature is represented as a vector $[\mathbf{s}_i]$ of dimension 6 using the inverse depth parametrization proposed by Javier Civera in [31]. This parametrization uses six parameters to define the position of a feature in a 3-Dimensional space. Each feature is defined by the position of a point (x_0, y_0, z_0) where the camera first saw the feature, the direction of a line based on that point and the inverse distance from the point to the feature along the line. This reference system allows the initialization of the features without any prior knowledge about the scene. This is important in exterior scenes where features with very different depths can coexist.

$$s_i = [x_0, y_0, z_0, \theta, \phi, \rho] \tag{17}$$

6.2 Prediction and Correction Stages

Extended Kalman filter (EKF) is used to implement the main algorithm loop, which has two stages: prediction and correction. In the prediction stage, uncertainty is propagated using the movement model. The correction stage uses real and predicted measurements to compute a correction to the prediction stage. Both stages need a precise description of the stochastic variables involved in the system.

There are mainly two approaches to implement this filter: extended Kalman filter and particle filter (FastSLAM). Both filters use the same formulation of the problem but have different approaches to the solution. The advantages of the Kalman filter are the direct estimation of the covariance matrix and the fact that it is a closed mathematical solution.

Its disadvantages are the increase of computational requirements as the number of features increase, the need of the model's linearization and the assumption of gaussian noise. On the other hand, particle filters can deal with non-linear,

non-gaussian models, but the solution they provide depends on an initial random set of particles which can differ in each execution. Prediction stage is formulated using linear equations

$$\begin{aligned}\hat{X}_{k+1} &= A \cdot X_k + B \cdot U_k \\ \hat{P}_{k+1} &= A \cdot P_k \cdot A^T + Q\end{aligned} \tag{18}$$

Where A is the transition matrix, B is the control matrix and Q is the model's covariance. Camera movement is modeled using a constant velocity model. Accelerations are included in a random noise component. For a variable n, which represents any of the position components (x, y, z) or the rotation components (p, q, r), we have:

$$n_{k+1} = n_k + v_k \cdot \Delta t \tag{19}$$

Where v_k is the derivative of n. We can estimate v_k as the differences in position,

$$n_{k+1} = n_k + \left(\frac{n_k - n_{k-1}}{\Delta t}\right)\Delta t = 2n_k - x_{n-1} \tag{20}$$

Feature movement is considered constant and therefore is modeled by an identity matrix. Now, full state model can be constructed

$$\begin{bmatrix} x_{k+1} \\ x_k \\ y_{k+1} \\ y_k \\ z_{k+1} \\ z_k \\ r_{k+1} \\ r_k \\ p_{k+1} \\ p_k \\ q_{k+1} \\ q_k \\ s_{1,k+1} \\ \cdots \end{bmatrix} = \begin{bmatrix} 2 & -1 & & & & & & & & \\ 1 & 0 & & & & & & & & \\ & & 2 & -1 & & & & & & \\ & & 1 & 0 & & & & & & \\ & & & & 2 & -1 & & & & \\ & & & & 1 & 0 & & & & \\ & & & & & & 2 & -1 & & \\ & & & & & & 1 & 0 & & \\ & & & & & & & & 2 & -1 \\ & & & & & & & & 1 & 0 \\ & & & & & & & & & \mathbf{I} \\ & & & & & & & & & & \ddots \end{bmatrix} \begin{bmatrix} x_k \\ x_{k-1} \\ y_k \\ y_{k-1} \\ z_k \\ z_{k-1} \\ r_k \\ r_{k-1} \\ p_k \\ p_{k-1} \\ q_k \\ q_{k-1} \\ s_{1,k} \\ \cdots \end{bmatrix} \tag{21}$$

Correction stage uses a non-linear measurement model. This model is the pin-hole camera model. The formulation of the Extended Kalman Filter in this scenario is

$$\begin{aligned} K_k &= \hat{P}_k \cdot J^T \left(J \cdot P \cdot J^T + R\right)^{-1} \\ X_k &= \hat{X}_k + K_k \cdot \left(Z_k - H\left(\hat{X}_k\right)\right) \\ P_k &= \hat{P}_k - K_k \cdot J \cdot \hat{P}_k \end{aligned} \tag{22}$$

where Z_k is the measurement vector, $H(X)$ is the non-linear camera model, J is the jacobian of the camera model and K_k is the Kalman gain.

The movement of the system is modeled as a solid with constant motion. Acceleration is considered a perturbation to the movement. A pin-hole camera model is used as a measurement model.

$$\begin{bmatrix} \lambda u \\ \lambda v \\ \lambda \end{bmatrix} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot [R|T] \cdot \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} \tag{23}$$

where u and v are the projected feature's central coordinates and λ is a scale factor. Distortion is considered using a four parameter model (k1, k2, k3, k4)

$$\begin{aligned} r^2 &= u^2 + v^2 \\ C_{dist} &= 1 + k_0 r^2 + k_1 r^4 \\ x_d &= u \cdot C_{dist} + k_2 (2u \cdot v) + k_3 \left(r^2 + 2u^2\right) \\ y_d &= v \cdot C_{dist} + k_2 \left(r^2 + 2v^2\right) + k_3 (2u \cdot v) \end{aligned} \tag{24}$$

State error covariance matrix is initialized in a two-part process. First, elements related to the position and orientation of the camera, $\mathbf{x}$, are initialized as zero or as a diagonal matrix with very small values. This represents that the position is known, at the first instant, with very low uncertainty. The initialization of the values related to the features, $\mathbf{s}_i$, must be done for each feature seen for first time. This initialization is done using the results from

$$\mathbf{P}_{k|k}^{new} = J \begin{bmatrix} \mathbf{P}_{k|k} & & \\ & \mathbf{R}_i & \\ & & \sigma_\rho^2 \end{bmatrix} J^T \tag{25}$$

where

$$\mathbf{J} = \begin{bmatrix} I & & 0 & 0 \\ \frac{\partial \mathbf{s}}{\partial \mathbf{xyz}} \; \frac{\partial \mathbf{s}}{\partial \mathbf{pqr}} & 0\ 0 \cdots & \frac{\partial \mathbf{s}}{\partial x_d, y_d} & \frac{\partial \mathbf{s}}{\partial \rho_0} \end{bmatrix} \tag{26}$$

$$\frac{\partial \mathbf{s}}{\partial \mathbf{xyz}} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}; \frac{\partial \mathbf{s}}{\partial \mathbf{pqr}} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ \frac{\partial \theta}{\partial p} & \frac{\partial \theta}{\partial q} & \frac{\partial \theta}{\partial r} \\ \frac{\partial \phi}{\partial p} & \frac{\partial \phi}{\partial q} & \frac{\partial \phi}{\partial r} \\ 0 & 0 & 0 \end{bmatrix}; \frac{\partial \mathbf{s}}{\partial x_d,\, y_d} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ \frac{\partial \theta}{\partial x_d} & \frac{\partial \theta}{\partial y_d} \\ \frac{\partial \phi}{\partial x_d} & \frac{\partial \phi}{\partial y_d} \\ 0 & 0 \end{bmatrix}; \frac{\partial \mathbf{s}}{\partial \rho_0} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \tag{27}$$

Taking into account that a robust feature tracking and detection is a key element in the system, a Mahalanobis' test is used in order to improve the robustness of feature matching. The filter is implemented using Mahalanobis' distance between the predicted feature measurement and the real measurement. Mahalanobis' distance weighs Euclidean distance with the covariance matrix. This distance is the input to a χ^2 test which rejects false matches.

$$(Z - J \cdot X)^t \cdot C^{-1} (Z - J \cdot X) > \chi_n^2 \tag{28}$$

where

$$C = H \cdot P \cdot H^T + R \tag{29}$$

Finally, it should be noted that the reconstruction scale is an unobservable system state. This problem is dealt with using inverse depth parametrization which avoids the use of initialization features of known 3D positions. This permits the use of the algorithm in any video sequence. Without these initialization features, the problem becomes dimensionless. The scale of the system can be recovered using the distance between two points or the position between the camera and one point. Computational cost is dependant on the number of features in the scene, and so an increase in the scene's complexity affects processing time in a negative way. Robust feature selection and matching are very important to the stability of the filter and a correct mapping. Experiments carried out successfully were made offline on sequences taken from the UAV.

7 Experimental Application and Tests

7.1 Visual Tracking Experiments

Tracking algorithms are fundamental to close the vision control loop in order to give an UAV the capability to follow objects. Hence, it is important to ensure the reliability of the tracker. Some experiments were conducted on images taken on test flights. Such experiments, where interest points were extracted with the Harris algorithm and tracked with the Lukas–Kanade algorithm, have proven to be fast enough so as to close the control loop at 17 Hz. However, if there are too many features selected to represent an object, the algorithm's speed slows down because of the calculation of the image derivatives.

SIFT features are very robust and rely on the advantage that the matching process does not depend on the proximity of two consecutive frames. On the other hand, the computational cost of the extraction is expensive. For that reason, they are suitable for visual servoing only if the displacements of the helicopter are forced to be very slow in order to avoid instabilities when closing the loop.

Tracking based on appearance proves to be very fast and reliable for acquired sequences at frame rates above 25 fps. This procedure is very sensitive to abrupt changes in the position of the tracked patch as long as the number of parameters of the motion model is higher than 3. This can be solved using stacks of trackers, each of which must have a different warping function that provides an estimation of the parameter to the next level of the stack. Simple warping functions give an estimation of more complex parameters. In the case of a simple tracker the translation-only warping function is the most stable one. Figure 10a shows the evolution of the parameters in a sequence of 1,000 images, and Fig. 10b the SSD error between the template image and the warped patch for each image.

7.2 Visual Servoing Experiments

The basic idea of visual servoing is to control the position of the helicopter based on an error in the image, or in a characteristic extracted from the image. If the

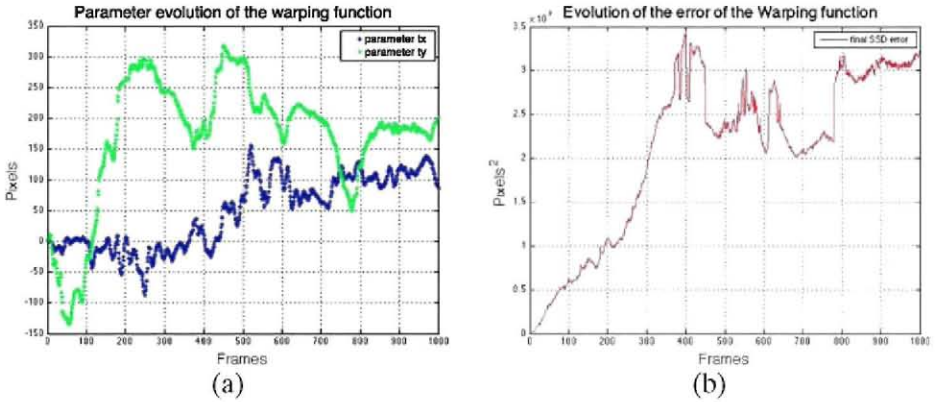


Fig. 10 Evolution of the translation parameter during the tracking process of a patch along 1,000 frames (**a**). **b** shows the SSD error of warped patch with respect to the template

control error is in the image plane, the measure of the error is a vector (in pixels) that represents the distance from the image's center to the feature's position. Figure 11 shows the basic idea of the error and the 2D visual servoing. In this sense, there are two ways to use this error in different contexts. One approach is to track features that are static in the scene. In this case, the control tries to move the UAV to align the feature's position with the image's center by moving the helicopter in the space.

Vision-based references are translated into helicopter displacements based on the tracked features. Velocity references are used to control the UAV, so that when the feature to track changes—as happens, for example, when another window of a building is chosen—velocity references change in order to align the UAV with the window.

The displacement of the helicopter when it tries to align with the feature being tracked is displayed in Fig. 12a. Vertical and lateral displacements of the helicopter are the consequence of the visual references generated from the vertical and horizontal positions of the window in the image. Figure 12b shows the displacement of the helicopter when the window above displayed was tracked, and Fig. 13 shows the velocity references when another window is chosen.

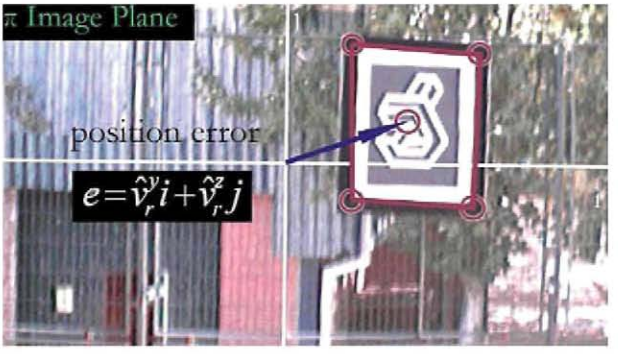
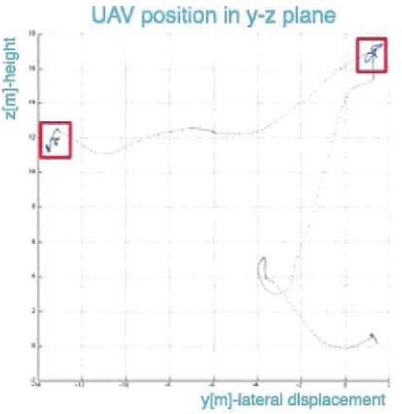


Fig. 11 Error measure in 2D visual servoing consists in the estimation of the distance of the reference point to the image's center

(a)



(b)

Fig. 12 Window being tracked during a visual servoing task (**a**), in which the UAV's vertical and lateral displacements are controlled by the visual control loop in order to fix the window in the center of the image, while the approaching movement is controlled by the GPS position controller. **b** Shows UAV vertical and lateral positions during the visual controlled flight. After taking off, the UAV moves to two positions (*marked with the red rectangles*) in order to consecutively track two external visual references that consist of two different windows

Another possible scenario is to keep the UAV hovering and to track moving objects in the scene. Experiments have been conducted successfully in order to proof variation of the method with good results. Control of the camera's Pan-Tilt Platform using 2D image servoing tries to keep a moving object in the image's center. In this case, position references are used instead of velocity in order to control the camera's pan and tilt positions. Figure 14 shows a car carrying a poster being tracked by moving the camera's platform.

Fig. 13 Velocity references change when a new feature is selected, in this case when another window is selected as shown in Fig. 12. Visual control takes the feature to the image center

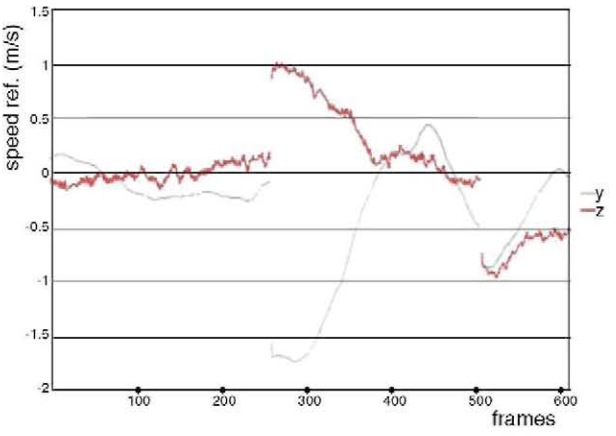


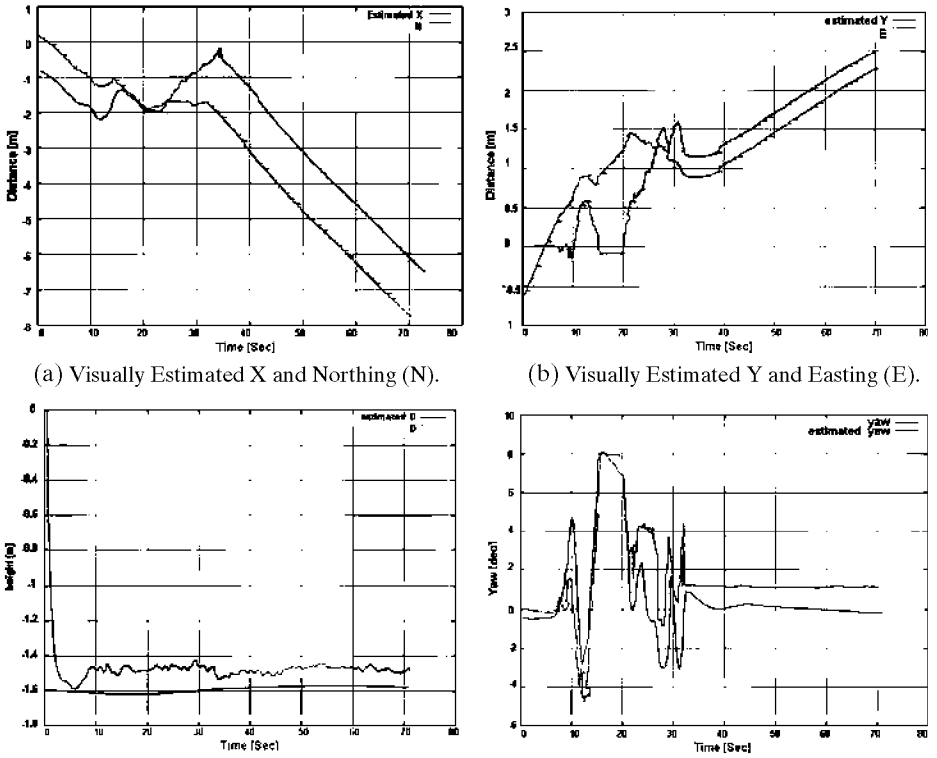
Fig. 14 Tracking of moving object. Servoing is perform on the pan-tilt platform. Notice that velocity in the cartesian coordinates is 0.0 (each component is printed on the image) since the UAV is hovering. Tracking is performed using corner features as explained in Section 3.2

7.3 Height and Motion Estimation Using a Stereo System

Stereo tests are made using a Firewire stereo system camera onboard the UAV. In these experiments, the helicopter is commanded to fly autonomously following a given trajectory while the onboard stereo vision algorithm is running. The experiments find the correlation between the stereo visual estimation and the onboard helicopter state given by its sensor suite. Figure 15 shows the results of one flight trial in which the longitudinal displacement (X), lateral displacement (Y), altitude (H) and relative orientation are estimated. Altitude is computed negative since the helicopter's body frame is used as a reference system. Each estimation is correlated with its similar value taken from the onboard helicopter state, which uses an EKF to fuse onboard sensors. Table 2 shows the error analysis based on the mean square error of the visual estimation and the helicopter's state. Four measures of the mean squared error are used: the error vision-GPS Northting (MSE_N^V), the error vision-GPS Easting (MSE_E^V), the error vision-yaw (MSE_ψ^V) and the error vision-altitude (MSE_H^V).

7.4 Power Lines Inspection

Besides visual servoing and image tracking applications, other experiments have been conducted to achieve object recognition in inspection tasks. Major contributions and successful tests were obtained in power lines' inspection. The objective of the application developed at the computer vision group is to identify powered lines and electrical isolators. The methodology that has been employed is based on the Hough transform and on Corner detectors that find lines in the image that are associated with the catenary curve formed by the hanging wire. Interest points are used to locate the isolator. Once both components are detected in the image, tracking can be initiated to make close up shots with the appropriate resolution needed for expert inspection and detection of failures. Figure 16 shows images of the UAV



(a) Visually Estimated X and Northing (N). (b) Visually Estimated Y and Easting (E).

(c) Visually Estimated H and helicopter altitude. (d) Visually Estimated Yaw and helicopter Yaw.

Fig. 15 Results using a stereo system. Four parameters are estimated for this experiment: the longitudinal displacement (X) (**a**), the lateral displacement (Y) (**b**), altitude (H) (**c**) and relative orientation (yaw) (**d**)

approaching a power line while in the sub-image the onboard camera displays the detection of the line and the isolator.

Stereo System has also been used to estimate the UAV distance and altitude with respect to power lines. In these tests, the line is detected using the Hough Transform. If the camera's angles, stereo system calibration and disparity are known, it is possible to determine the position of the helicopter relative to the power line. Some tests using the Stereo system onboard the helicopter were carried out to obtain the distance to the power line from the helicopter. The power Line is detected using Hough transform in both images. In this test, the helicopter was initially 2 m below the power line. Afterwards, it rises to be at the same altitude of the cable and then

Table 2 Error analysis for the helicopter's experimental trials

Exp.	Test
MSE_N^V **m**	1.0910
MSE_E^V **m**	0.4712
MSE_ψ^V **deg**	1.7363
MSE_H^V **m**	0.1729

Fig. 16 Power line and Isolator detection using the UAV vision system

it returns to its original position. Figure 17 shows the distance and height estimated from the UAV to the power line during this test. Additional tests can be seen on the Colibri Project's Web Page

7.5 Mapping and Positioning using Visual SLAM

The SLAM algorithm explained in Section 6 is used in a series of image sequences of trajectories around a 3D scene that were performed flying in autonomous mode navigation based on way points and desired heading values. The scene is composed of many objects, including a grandstand, a van and many other elements, and also of a series of marks feasible for features and corners' detection. For each flight test, a 30 fps image sequence of the scene was obtained, associating the UAV attitude information for each one. That includes the GPS position, IMU data (Heading, body frame angles and displacement velocities) and the helicopter's position, estimated by the Kalman filter on the local plane with reference to takeoff point. Figure 18 shows a reconstruction of one flight around one scene test.

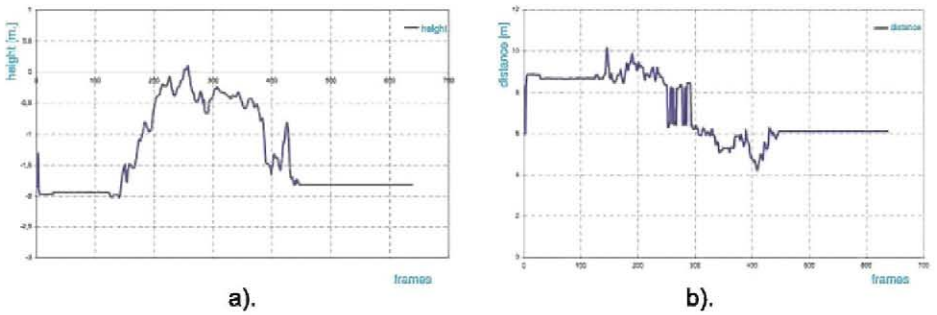


Fig. 17 Distance and height estimation to the power lines using a stereo system onboard the UAV

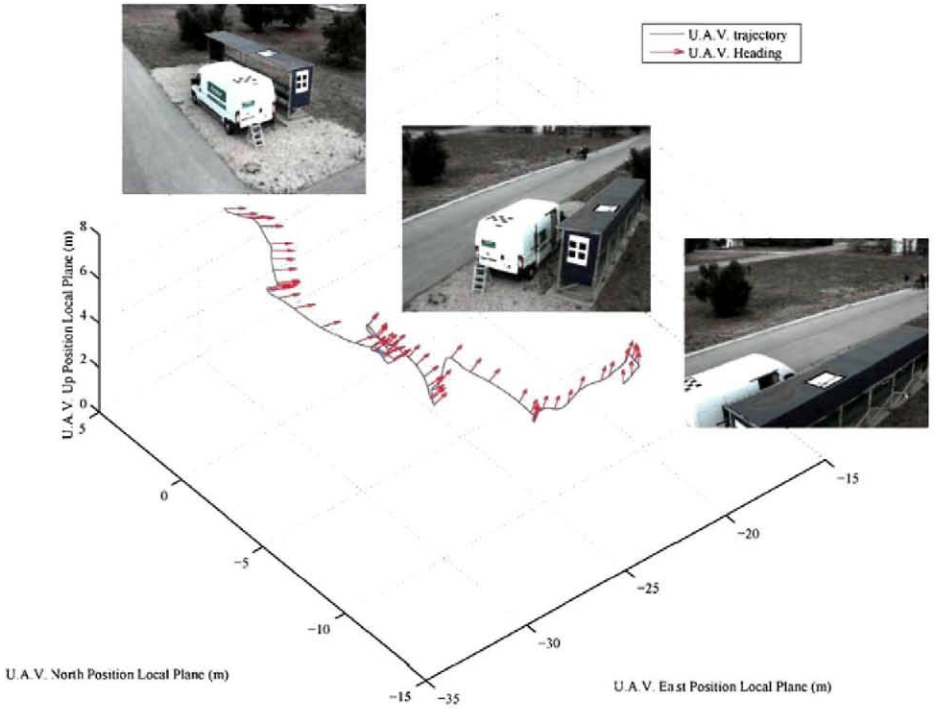


Fig. 18 Three-dimensional flight trajectory and camera position reconstruction obtained using the flightlog data. The *blue line* depicts the translational movement and the *red arrows* represent the heading direction of the camera (pitch and yaw angles). *Superimposed images* show the different perspectives obtained during the flight sequence around the semi-structured scene

Results for tests using a tracking algorithm for scene elements are shown on Fig. 19a. Reconstructed features are shown as crosses. In the figure, some reference planes were added by hand in order to make interpretation easier. Figure 19b shows an image from the sequence used in this test.

Results show that the reconstruction has a coherent structure but that the scale of the reconstruction is function of the initialization values. The scale can be recovered using the distance between two points or the positions of one point and the camera.

The camera movement relative to the first image is compared with the real flight trajectory. For this, the (x, y, z) axis on the camera plane are rotated so that they are coincident with the world reference plane used by the UAV. The heading or yaw angles (ψ) and the Pitch angle (θ) of the helicopter, in the first image of the SLAM sequence, define the rotational matrix used to align the camera and UAV frames.

The displacement values obtained using SLAM are rotated and then scaled to be compared with the real UAV trajectory. Figure 20 shows the UAV and SLAM trajectories and the medium square error (MSE) between real flight and SLAM displacement for each axe. The trajectory adjusts better to the real flight as the features reduce their uncertainty, because the more images are processed, more measurements refine features estimation.

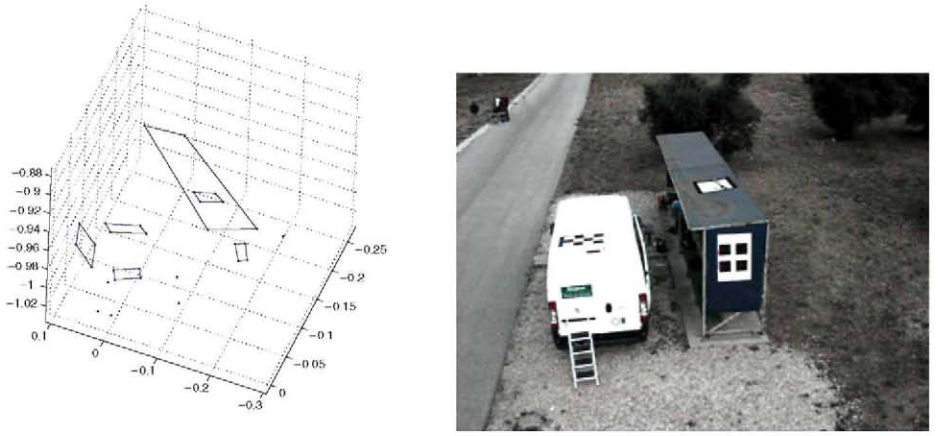


Fig. 19 Scene reconstruction. The *upper figure* shows reconstructed points from the scene shown in the *lower figure*. *Points* are linked manually with lines to ease the interpretation of the figure

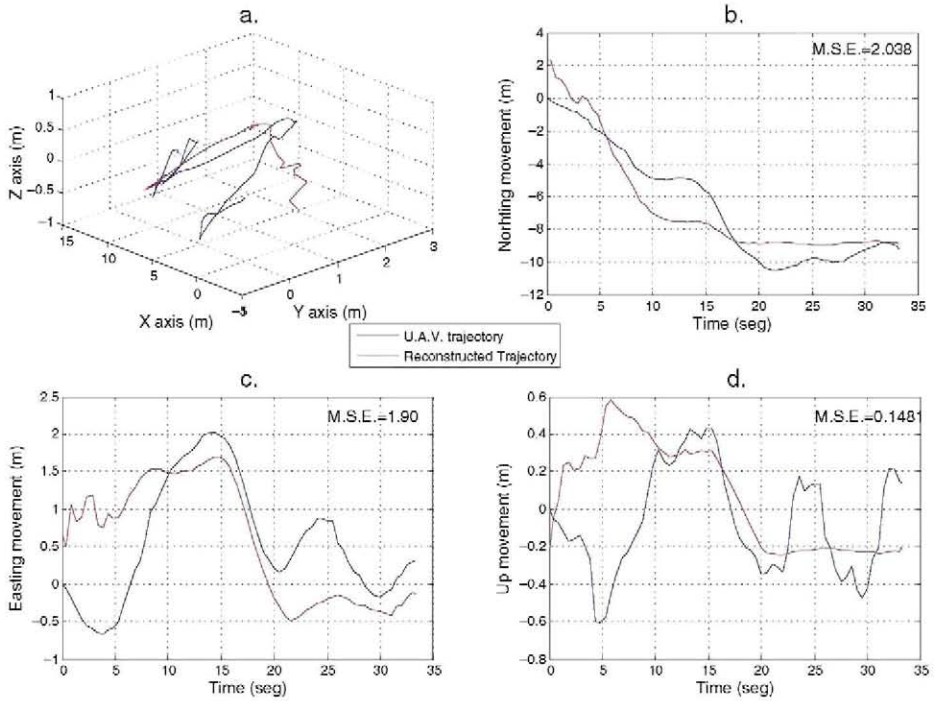


Fig. 20 SLAM reconstructed trajectory vs. UAV trajectory. **a** Three-dimensional flight, **b** north axe in meters, **d** east axe in meters, **c** altitude in meters. The reconstructed trajectory adjusts best to the real flight as soon as more images are processed and the uncertainty of the features is thus reduced

8 Conclusions

This paper dealt with the researches, results and discussion of the use of several techniques of computer vision onboard an UAV. These computer vision techniques are not merely used for acquiring environmental visual information that can be used afterwards by off-line processing. That's why the paper also shows how computer vision can play an important role on-line during the flight itself in order to acquire the adequate sequences necessary to actively track targets (fixed or moving ones) and to guide and control flight trajectories.

Image processing algorithms are very important, and are often designed to detect and track objects along the sequences, whether key points are extracted by the algorithm itself or are externally determined visual targets. Successful, wide spread algorithms onboard an UAV have test bed challenges and thus provide a source of inspiration for their constant improvement and for achieving their better robustness. Some of those test bed challenges are the non-structured and changing light conditions, the highly vibrating and quick and sharp movements, and on-line requirements when necessary.

Some improvements have been presented and tested in the following two types of image processing algorithms: feature tracking and appearance-based tracking, due to the above mentioned characteristics. When using the SIFT key point detector, the algorithm reduces and classifies the key points for achieving a more robust and quick tracking as stated in Section 3. When tracking a whole visual target, an ICA based algorithm is used in a multi-scale hierarchical architecture that makes it robust for scaling. In both type of algorithms, a Kalman filter has been implemented in order to improve the consistence of the features and targets' movements within the image plane, a feat that is particularly relevant in quick changing sequences, as stated in Section 3.3.

The filtered outputs of the image processing algorithms are the visual measurements of the external references that, when compared to their desired position, are introduced in a decoupled position control structure that generates the velocity references in order to control the position of the UAV according to those external visual references. Depending on the type of information extracted by the image processing algorithms (i.e. bi-dimensional translation, rotation, 3D measurements, among others), the UAV's position and orientation control can be a mix of visual based control for some UAV coordinates and GPS based control for some others. A Kalman filter can also be computed in future developments to produce unified UAV estimation and control based on visual, GPS, and inertial information.

This paper also shows that it is possible to obtain robust and coherent results using Visual SLAM for 3D mapping and positioning in vague structured outdoor scenes from a mini UAV. The SLAM algorithm has been implemented using only visual information without considering any odometric or GPS information. Nonetheless, this information has been later used in order to compare and evaluate the obtained results. The state of the system comprises a 12 variable array (position, orientation and their rates), where the inverse depth parametrization has been used in order to avoid the initialization of the distances to the detected visual features, that otherwise becomes a drawback when using SLAM outdoors in unknown environments. The rest of the state array is made up of the tracked features, being ten the minimum allowed number. The prediction stage in EKF has been modeled considering constant

velocity for both the position-orientation coordinates and the feature movements in the image plane. The correlation stage in the EKF uses a non-linear camera model that includes a pin-hole distortion model for the sake of more accurate results. Within the implemented SLAM algorithm the Mahalanobis' distance is used to discharge far away matched pairs that can otherwise distort the results.

Based on the results of our work, we conclude that the UAV field has reached an important stage of maturity in which the possibility of using UAVs in civilian applications is now imaginable and in some cases attainable. We have experimentally demonstrated several capabilities that an autonomous helicopter can have by using visual information such as navigation, trajectory planning and visual servoing. The successful implementation of all these algorithms confirms the necessity of dotting UAVs with additional functionalities when tasks like outdoor structures' inspection and object tracking are required.

Our current work is aimed at increasing these capabilities using different visual information sources like catadioptric systems and multiple view systems, and extending them to 3D image-based visual servoing, where the position and orientation of the object will be used to visually conduct the helicopter. The challenge is to achieve real-time image processing and tracking algorithms to reduce the uncertainty of the measure. The field of computer vision for UAVs can be considered as a promising area for investing further research for the benefit of the autonomy and applicability of this type of aerial platforms, considering that reliability and safety have become major research issues of our community.

References

Puri, A., Valavanis, K.P., Kontitsis, M.: Statistical profile generation for traffic monitoring using real-time UAV based video data. In: Control and Automation, 2007. MED '07. Mediterranean Conference on, MED, pp. 1–6 (2007)

Nikolos, I.K., Tsourveloudis, N.C., Valavanis, K.P.: Evolutionary algorithm based path planning for multiple UAV cooperation. In: Advances in Unmanned Aerial Vehicles, Intelligent Systems, Control and Automation: Science and Engineering, pp. 309–340. Springer, The Netherlands (2007)

Nikolos, I.K., Tsourveloudis, N.C., Valavanis, K.P.: A UAV vision system for airborne surveillance. In: Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on, pp. 77–83. New Orleans, LA, USA (2004), May

Nikolos, I.K., Tsourveloudis, N.C., Valavanis, K.P.: Multi-UAV experiments: application to forest fires. In: Multiple Heterogeneous Unmanned Aerial Vehicles, Springer Tracts in Advanced Robotics, pp. 207–228. Springer, Berlin (2007)

Green, W., Oh, P.Y.: The integration of a multimodal mav and biomimetic sensing for autonomous flights in near-earth environments. In: Advances in Unmanned Aerial Vehicles, Intelligent Systems, Control and Automation: Science and Engineering, pp. 407–430. Springer, The Netherlands (2007)

Belloni, G., Feroli, M., Ficola, A., Pagnottelli, S., Valigi, P.: Obstacle and terrain avoidance for miniature aerial vehicles. In: Advances in Unmanned Aerial Vehicles, Intelligent Systems, Control and Automation: Science and Engineering, pp. 213–244. Springer, The Netherlands (2007)
Dalamagkidis, K., Valavanis, K.P., Piegl, L.A.: Current status and future perspectives for unmanned aircraft system operations in the US. In: Journal of Intelligent and Robotic Systems, pp. 313–329. Springer, The Netherlands (2007)
Long, L.N., Corfeld, K.J., Strawn, R.C.: Computational analysis of a prototype martian rotorcraft experiment. In: 20th AIAA Applied Aerodynamics Conference, number AIAA Paper 2002–2815, Saint Louis, MO, USA. Ames Research Center, June–October 22 (2001)
Yavrucuk, I., Kanan, S., Kahn, A.D.: Gtmars—flight controls and computer architecture. Technical report, School of Aerospace Engineering, Georgia Institute of Technology, Atlanta (2000)
Buenaposada, J.M., Munoz, E., Baumela, L.: Tracking a planar patch by additive image registration. In: Proc. of International Workshop, VLBV 2003, vol. 2849 of LNCS, pp. 50–57 (2003)
Miller, R., Mettler, B., Amidi, O.: Carnegie mellon university's 1997 international aerial robotics competition entry. In: International Aerial Robotics Competition (1997)
Montgomery, J.F.: The usc autonomous flying vehicle (afv) project: Year 2000 status. Technical Report IRIS-00-390, Institute for Robotics and Intelligent Systems Technical Report, Los Angeles, CA, 90089-0273 (2000)
Saripalli, S., Montgomery, J.F., Sukhatme, G.S.: Visually-guided landing of an unmanned aerial vehicle. IEEE Trans. Robot Autom. **19**(3), 371–381, June (2003)
Mejias, L.: Control visual de un vehiculo aereo autonomo usando detección y seguimiento de características en espacios exteriores. PhD thesis, Escuela Técnica Superior de Ingenieros Industriales. Universidad Politécnica de Madrid, Spain, December (2006)
Mejias, L., Saripalli, S., Campoy, P., Sukhatme, G.: Visual servoing approach for tracking features in urban areas using an autonomous helicopter. In: Proceedings of IEEE International Conference on Robotics and Automation, pp. 2503–2508, Orlando, FL, May (2006)
Mejias, L., Saripalli, S., Sukhatme, G., Campoy, P.: Detection and tracking of external features in a urban environment using an autonomous helicopter. In: Proceedings of IEEE International Conference on Robotics and Automation, pp. 3983–3988, May (2005)
Mejias, L., Saripalli, S., Campoy, P., Sukhatme, G.: Visual servoing of an autonomous helicopter in urban areas using feature tracking. J. Field Robot. **23**(3–4), 185–199, April (2006)
Harris, C.G., Stephens, M.: A combined corner and edge detection. In: Proceedings of the 4th Alvey Vision Conference, pp. 147–151 (1988)
Lowe, D.G.: Distintive image features from scale-invariant keypoints. Int. J. Computer Vision **60**(2), 91–110 (2004)
Duda, R.O., Hart, P.E.: Use of the hough transformation to detect lines and curves in pictures. Commun. ACM **15**(1), 11–15 (1972)
Canny, J.: A computational approach to edge detection. IEEE Trans. Pattern Anal. Machine Intel. **8**(6), 679–698, November (1986)
Feldman, G., Sobel, I.: A 3×3 isotropic gradient operator for image processing. Presented at a talk at the Stanford Artificial Project (1968)
Mejías, L., Mondragón, I., Correa, J.F., Campoy, P.: Colibri: Vision-guided helicopter for surveillance and visual inspection. In: Video Proceedings of IEEE International Conference on Robotics and Automation, Rome, Italy, April (2007)
Lucas, B.D., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: Proc. of the 7th IJCAI, pp. 674–679, Vancouver, Canada (1981)
Beis, J.S., Lowe, D.G.: Shape indexing using approximate nearest-neighbour search in high-dimensional spaces. In: CVPR '97: Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition (CVPR '97), p. 1000. IEEE Computer Society, Washington, DC, USA (1997)
Fischer, M.A., Bolles, R.C.: Random sample concensus: a paradigm for model fitting with applications to image analysis and automated cartography. Commun. ACM **24**(6), 381–395 (1981)
Baker, S., Matthews, I.: Lucas-kanade 20 years on: A unifying framework: Part 1. Technical Report CMU-RI-TR-02-16, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, July (2002)
Mejias, L., Campoy, P., Mondragon, I., Doherty, P.: Stereo visual system for autonomous air vehicle navigation. In: 6th IFAC Symposium on Intelligent Autonomous Vehicles (IAV 07), Toulouse, France, September (2007)